International Journal of Mathematics
and Computer Applications Research (IJMCAR)
ISSN (P): 2249-6955; ISSN (E): 2249-8060
Vol. 8, Issue 5, Oct 2018, 13-24

Original Article

AN EFFICIENT MODIFIED CUCKOO SEARCH OPTIMIZATION WITH THE RATIONAL BARRIER FUNCTION

RANA Z. Al-KAWAZ[1], HUDA I. AHMED[2] & ABBAS Y. Al-BAYATI[3]

[1]*Department of Mathematics, College of Basic Education, University of Telafer, Mosul, Iraq*

[2]*Department of Operation Research and Intelligent Techniques, College of Computer Sciences and Mathematics, University of Mosul, Iraq*

[3]*University of Telafer, Mosul, Iraq*

ABSTRACT

Recent modern optimization algorithms are often very successful in solving NP-hard improvement problems. One of the main disadvantages of the search algorithm in cuckoo (CS) is the weakness of its ability as a local search algorithm and slow in its speed of convergence, which results in a limited access to the target. There are two categories that can be solved by this algorithm: First, the unconstrained functions are solved. Second, which is the focus of this article, is the constraint functions and how to deal with them using a rational formula of the Barrier function. The results show that the CRB (Cuckoo & Rational Barrier Function) algorithm has a better convergence velocity than the algorithm (CS) by observing the separation of the numerical results of the browser in which the accuracy of the new algorithm on some engineering functions.

KEYWORDS: Cuckoo (CS) Search, Local Search, Rational Barrier Function, Constrained Optimization & Optimization Algorithms

Received: Aug 11, 2018**; Accepted:** Aug 31, 2018**; Published:** Sep 24, 2018**; Paper Id.:** IJMCAROCT20182

1. INTRODUCTION

The main theme that led to the need to find intelligent search algorithms is to solve the problems that consist of multiple dimensions as an example of engineering problems. Most evolutionary optimization problems in the nonlinear engineering have many constraines. Metaheuristics algorithm, based on the swarm intelligence, is an important part of the global contemporary optimization algorithms. Good examples are the genetic algorithms ([12], [10]), a model developed from the Particle Swarm Optimization (PSO) [14] and Harmony Search algorithm (HS) [32]. A number of recently developed metaheuristics algorithms have been developed based on the intelligence of the swarm in nature ([14], [7]). Later, evolutionary optimization algorithms have gone a long way in speeding evolution, becoming a significant topic in improving the structure ([26], [30]). In 2009, both Yang and Deb [30] introduced the concept of the Cuckoo (CS) search algorithm, and defined it as a highly appropriate algorithm for evolution and In [31] the engineering side was introduced using the optimization method and the design of springs and welded beam structures. Day by day, new algorithms are being developed, demonstrating their strength and effectiveness. As an example, a comparison shown by the cuckoo algorithm with traditional algorithms yields superior performance. The advantages of the CS algorithm include, first: enable it to search the world well and second: the number of parameters used in the search is few. Gandomi et al. [9]. They determined that the best solutions they found using CS were the best solutions from solutions obtained through genetic

algorithms and PSO. The CS algorithm was widely used in structural design problems [14], aerodynamic improvements [17], and reliability improvement problems [27]. The reason for the slow speed of the CS algorithm is that it does not use gradient information. Which makes it difficult to apply this algorithm to complex engineering models where the cost of computation that exceeds the tolerance. In [16] the researchers developed and evaluated the performance of three algorithms for searching with Cuckoo based on increasing the conversion parameter between local and global random paths as the number of repetitions increased. The first algorithm has an increase in the linear switching coefficient, and the second parameter is applied to a steadily increasing conversion as the third parameter is used to convert the increasing energy.

2. THECUCKOO ALGORITHM AND BARRIER FUNCTION

2.1. The Cuckoo Search Algorithm

Now take a deep look at the cuckoo algorithm and explain its important details. Cuckoo birds behave currish they will use other bird nests to lay their eggs. These birds will be called birds of the cuckoo birds [19]. Host birds usually know when cuckoo eggs are in their nests either leave their eggs or abandon those nests. The cuckoo can adapt himself by placing eggs simulating host eggs in terms of colour and pattern. Giving them the ability to reproduce themselves and reduce the number of eggs disposed of. As for the ideal rules of the CS algorithm:

- The egg placed in each bird's nest is placed in a randomly selected nest each time.
- Future generations will continue with the best nests (good eggs).
- The egg developed by the cuckoo is inspected by host birds of a fixed number of nest, nests with the probability of $p_\alpha \in [0, 1]$. In this way, new random solutions (new nests) can replace host nests.

Let's discuss the rules above, (CS) have been implemented as follows. The solution is the whole egg in the nest. Thus, each cuckoo can place only one egg in a nest in its original form, although each nest can contain multiple eggs representing a range of solutions, in general. Therefore, there is the possibility that each bird and hatches, put only one egg in the nest in the same original form, giving us that each nest contains eggs of multiple qualities representing a range of solutions, as a general idea. Mathematically, these types of trivial problems of transformation are minimized to the maximum of problems with respect to the micro-equation (f (x)) = max (-f (x)). In contrast to the objective function, this function is now referred to as a fitness function [8].

Nature's axioms, animals search for food in a random or semi-random fashion. For random search, the animal search path is randomized without any discrimination because the next step depends on (location - current status - probability of moving to your next location). The chosen direction implicitly depends on the probability of the mathematical model [30]. From all of the above, the purpose of the CS algorithm can be written clearly is to look for x that reduces the function of the objective function f (x). Through these rules, we conclude that for the cuckoo nest i, the new generation solution can be defined within the k-th iteration as [30].

$$x_i^{k+1} = x_i^k + \alpha \oplus \text{Lévy}\ (\beta) \tag{1}$$

To illustrate the idea, we have to give each symbol a symbol:

α is the constant step of the method of Flying Lévy possible using its value α = 1.5.

⊕ Product code which means double entries.

After passing through the research on the behaviour of random search into the nature of animals with proof of behaviour characteristics meet Lévy flight. We can equate the random walk and the amount of repetition following by the Lévy flight using the distribution:

$$\text{Lévy} \sim u = k^{-\beta} \quad , \quad 0 < \beta \leq 3 \tag{2}$$

The variable u is certainly the normal distribution and the number of repetitions is k. How many times do we randomly generate these numbers, obey Lévy flights, and calculating the amount and direction of the step in a random manner using the Lévy distribution given in the two research papers [6], [18].

Which has (contrast with mean) infinite. Here, successive sequence / cuckoo steps are essentially the random walking process that corresponds to the energy distribution of the step length law with a heavy tail. We can say that if the cuckoo's egg is very similar to host eggs, it is unlikely to detect this coco egg, and therefore fitness must be linked to different solutions. Therefore, it is recommended to perform a random path in a biased manner with some random step sizes ([9], [30]). In Figure 1, in the Flowchart mode we show the Cuckoo search path to access the function formula:

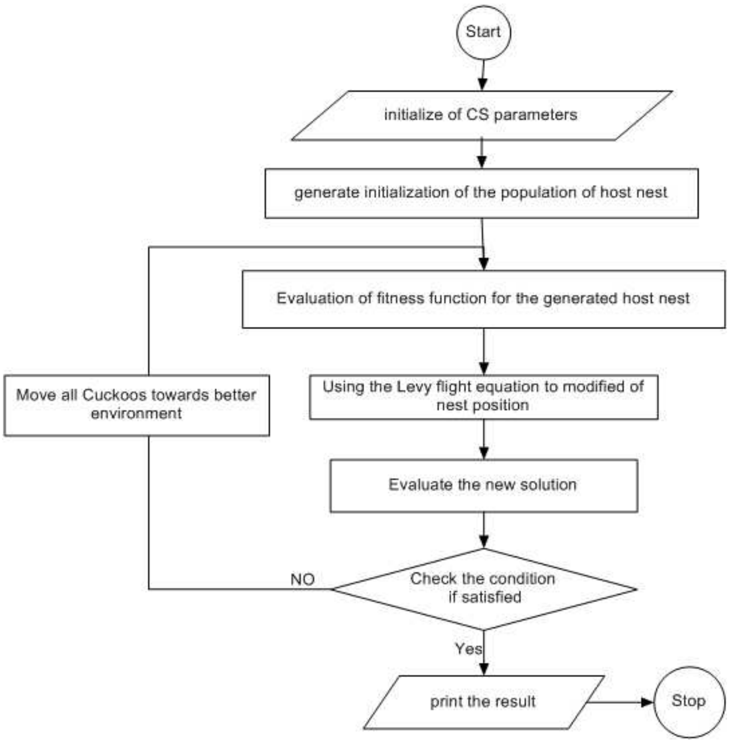

Figure 1: Illustrated the Cuckoo Search Algorithm [1]

One of the most important features of this algorithm is its simplicity. If we compare it with the rest of the algorithms that are categorized by metaheuristics algorithms based on population or proxy (PSO) and (HS), there is essentially only one parameter $p_{\propto}$ in CS (regardless of population size n). Therefore, very easy to implement. After speaking in general about the cuckoo algorithm and slow performance we give an introduction to some types of Barrier function and define the details of its formulas in the following section:

2.2. Barrier Function

The use of the SUMT technique has recently been popular, with the original restricted problem becoming a series of unrestricted problems. This was a natural approach because the non-codified reduction techniques were well developed. As research progressed in optimization algorithms, other methods were shown to be more efficient and reliable for "typical" optimization problems. However, the larger the problem, the more appropriate methods can become numerically ineffective. Thus, especially for structural optimization using rounding techniques, the new appearance of SUMT is appropriate [28].

The famous method in community optimization metaheuristic to deal with constraints is to use the Barrier function. The main idea of this method is to convert the problem of restricted improvement into an unrestricted problem by adding numerical coefficients for both the objective function and the constraints given to the question depending on the method used and the number and type of constraints to the problem. This technique, known as the Barrier method, is one of the famous methods of dealing with evolutionary algorithms. The current work has been using the same method [29].

2.3. Modified Frish's& Carroll's Barrier Functions

To view the nonlinear programming problem (NLP) with the inequality limitations, we give the following sample:

$$\text{minimize } f_0(x)$$

$$\text{subject to } c_i(x) \geq 0, i = 1, \dots, m \tag{3}$$

Where $x \in R^n, c_i(x): R^n \to R$, are real-valued functions. The classical (linear) Lagrangian of (NLP) is defined by:

$$L(x, u) = f_0(x) - \sum_{i=1}^{m} v_i c_i, \text{ i=1,…..,m} \tag{4}$$

There are factors that are important components in explaining the optimum conditions for NLP and the combination of algorithms to solve NLP. There are two forms of nonlinear functions that can be addressed as they are:

- The convex programming, through the writing of the saddle-point theorem based on the classical Lagrange method and the binary methods used, the solution L(x, vk) of the mini-vk can be updated.
- Nonconvex programming, L (x, vk) is often the most important form of a non-convex shape for vk, which is closer to v* and x in the vicinity of x*, where x (v*) is Kuhn-Tucker (NLP), which creates complexity in numerical issues.

It is possible to say that those who solved this question, the researcher Hestenes through his research [11] and Powell [23] Lagrangian extended questions of addition to the constraints of equality and Rockafellar [24], [25] (Equality and inequality), but the convex programming, Polyak and Teboulle [20] discussed a different class of Lagrange based on the Log-Sigmoid function. Polyak [22] then developed the NR method and gave the approach of the method in detail. In [2] they presented four types of hybrid structures for the function of Polyak's Log-Sigmoid with a firefly algorithm at a time of this year. As for nonconvex programming, a class of nonlinear langarians is usually given with the limitations of inequality to the question, which then gives us the unencumbered saddle points by the Mangasarian [15]. After looking at the lower pth function given by Charalambous [5]; the researcher Bertsekas [3] modified the exponential Lagrangian function as follows:

$$F(x,u,k) = f_0(x) + \frac{1}{\omega}\sum_{i=1}^{m} v_i \left[1 - e^{kc_i(x)}\right] \quad (5)$$

Polyak [21] gave two modified barrier functions, namely, modified Frish's function

$$F(x,u,k) = \begin{cases} f_0(x) - \frac{1}{\omega}\sum_{i=1}^{m} v_i \ln\left(1 + \omega c_i(x)\right), x \in \mathrm{int}S_k \\ +\infty, x \notin \mathrm{int}S_k \end{cases} \quad (6)$$

and modified Carroll's function:

$$C(x,u,k) = \begin{cases} f_0(x) - \frac{1}{\omega}\sum_{i=1}^{m} v_i (1 - [1 + \omega c_i(x)]^{-1}), x \in \mathrm{int}\ S_k \\ +\infty, x \notin \mathrm{int}\ S_k \end{cases} \quad (7)$$

where k > 0 is parameter and $S_k = \{x | 1 + \omega c_i(x) \geq 0, i = 1, \dots, m\}$.

In this article, we tried to make the most of these functions in (6) and (7). We proposed a fractional function consisting of a numerator and denominator as given in the following equation:

$$\frac{u \text{ Modified Frish's function}}{(1-u)\text{Modified Carroll's function}} = \text{New Rational Barrier Function} \quad (8)$$

Where is the scalar between 0<u<1.

3. MIXED CS AND RATIONAL BARRIER FUNCTION CRB

When dealing with general functions (linear - nonlinear), it is worth noting the difficulties in the question when finding the best solution for them and this is followed by formulas and research in finding the best ways. In this article, we combined two important types. The first type of treatment of the question and its transformation from a functionally restricted constraint and nonlinear constraints by the function of the barrier with the egg distribution method to the gums gave a good and appropriate formula for solving this type of issue. Take a look at Figure 2 we notice that the new algorithm method based on the division of two functions developed from the functions of the barrier (Frish's & Carroll's functions) and multiplied by a coefficient is the highest level at 1 and the least possible to take is the 0 in the significance and strength of the fractional formula, Cuckoo to get better behaviour and approach the optimal function of the problem when solving non-linear functions multiple constraints and complex engineering functions.:

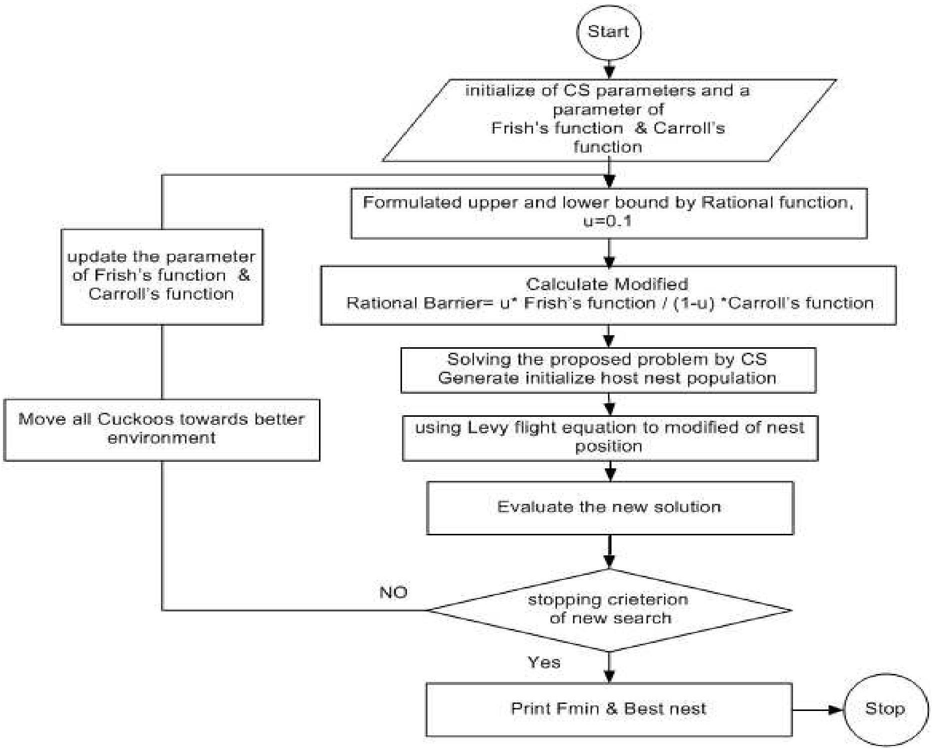

Figure 2: The New Algorithm CRB

4. STRUCTURAL ENGINEERING OPTIMIZATION

In this section we will show the efficiency of our new algorithm when compared with 15 nonlinear functions given in the Appendix within the article and a number of complex structural optimization problems, and sometimes there are no optimal solutions for attention. In order to find out how the CRB algorithm performs against the CS algorithm about 3 problems solve standard structural engineering testing.

Case 1: Spring Pressure

The geometric questions given in [4] consist of the following form and data, namely weight reduction for pressure pressure (Figure 3) controlled by minimum deviation limits, shear stress, frequency of mutation, outer diameter perimeter and design variables. There are three design variables: the X1 wire diameter, the average diameter of the X2 coil, and the number of active X3 files:

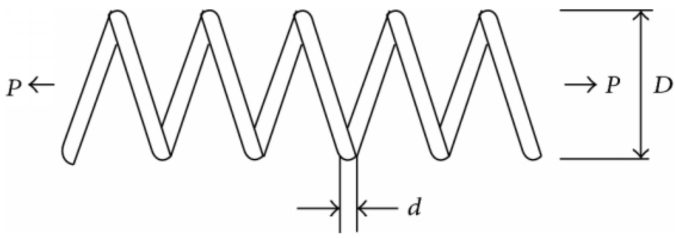

Figure 3: Spring Pressure

Case 2: Speed Reducer

The speed reduction question [4] shown in the Figure 4, with variables defined as follows:

After the face X_1, the tooth unit X_2, the number of teeth on the wing X_3, the length of the column 1 of the X_4 axillary, the length of column 2 between the bearings X_5 and diameter (column 1) X_6 and the diameter of the first X_7 (all variables are continuous except X_3 which is integer). The weight of the reducer should be reduced according to the gears of the gear teeth, the surface stress, the occasional deviations of the columns and the pressures in the column.

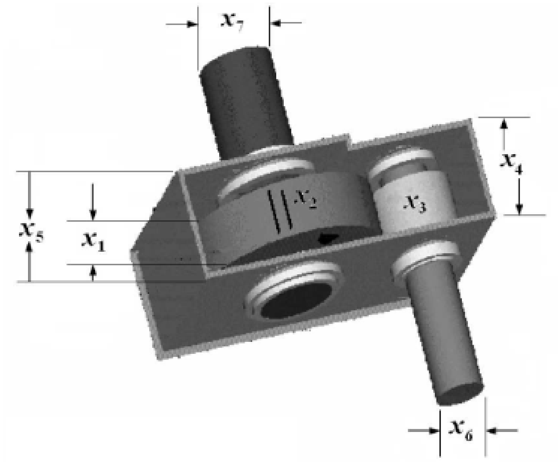

Figure 4: Speed Reducer

Case 3: Welded Beam

The last engineering questions are the welded package at the lowest cost [4]. Figure 5 shows the beam structure of the beam A and the weld required to load it to member B. The question is to find the minimum manufacturing cost with its four components:

X_1, X_2, X_3, X_4, τ, bending stress in the beam σ, buckling load on the bar Pc, and end deflection on the beam δ.

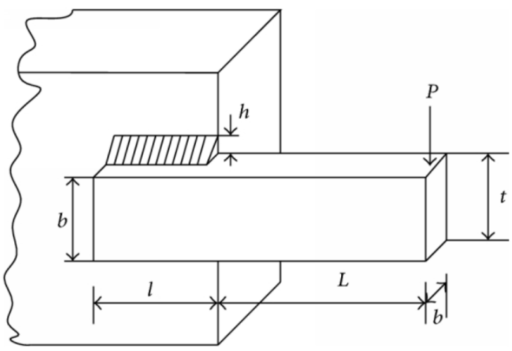

Figure 5: Welded Beam

The values in tables starting from Table 1 to Table 4 show the results of the best performance of the new algorithm in terms of the number of tools described in the tables below:

Table 1: Comparing CS Algorithm against CRB in 2D models for Number of nests=25 (Total Number of Iterations=10000, Time=2000)

Function	CS		CRB	
	fmin	Mean Best Nest	fmin	Mean Best Nest
F1	1.400000000000000e+010	3.179203962030118	1.380687544362068e+003	1.792844905347926
F2	1.500000000000000e+010	5.155094546754263	1.000000000000000e+010	3.643501189377146
F3	1.000000000000000e+010	2.104849965082111	0.199946282778281	1.599989645305540
F5	1.300000000000000e+010	5.372292496848714	1.000000000000000e+010	3.229702777984627
F6	6.784291464185811e+002	2.169880798217203	1.316848978579592e+002	1.016848978579592e+002
F7	1.000000000000000e+010	2.810772173507388	0.372381895049205	1.120287140654425
F8	1.000000000000000e+010	2.506881169340502	0.876789754634672	1.406630442441339
F10	1.209992957493290	0.799984915177992	0.209984423794242	0.499989304529111
F11	1.300000000000000e+010	1.095447189578642	1.000000000000000e+010	0.127130574560910
F12	1.000000000000000e+010	2. 475735361799080	1.000000000000000e+010	2.070593645908076
F13	1.156261555600451	1.937498420682251	0.156256259390069	0.937499998704024
F14	1.000000000000000e+010	0.671277061080557	7.121486997773123e+002	0.112499978892667
F15	3.229286417411150	0.945858903815962	0.262572203391223	0.189812314433039

Table 2: Comparing CS Algorithm against CRB in 3D and Engineering Models for Number of Nests=25 (Total Number of Iterations=10000, Time=2000)

Function	CS		CRB	
	fmin	Mean Best Nest	fmin	Mean Best Nest
F4	1.200000000000000e+010	2.957017834544342	1.000000000000000e+010	0.985248894239120
F9	1.900000000000000e+010	3.992116826916045	1.602811434499159	3.766666666666667
Spring	1.000000000000000e+010	5.361519719525025	0.801571652220139	0.737997242107562
Speed	3.2000000000000e+010	7.832525695537344	2.362038817325554e+003	6.185714285714285
Beam	1.724852313132572	3.229642967361158s	0.309403299807856	0.100000000000000

Table 3: Comparing CS Algorithm Against CRB in 2D Models for Number of Nests=25 (Total Number of Iterations=5000, Time=100)

Function	CS		CRB	
	fmin	Mean Best Nest	fmin	MEAN Best Nest
F1	1.400000000000000e+010	3.179203962030118	1.232828996529320e+008	1.222217898395526
F2	1.500000000000000e+010	5.155094546754263	1.141124068742142e+008	1.026826803301498
F3	1.000000000000000e+010	2.104849965082111	1.000000000000000e+010	0.906582056092352
F5	1.300000000000000e+010	5.372292496848714	1.000000000000000e+010	4.577859246219106
F6	6.784291464185811e+002	2.169880798217203	1.000000000000000e+010	1.898848112696566
F7	1.000000000000000e+010	2.810772173507388	0.837913891472630e+009	1.380030554991092
F8	1.000000000000000e+010	2.506881169340502	1.000000000000000e+010	2.322684997789185
F10	1.209992957493290	0.799984915177992	1.130633100672871e+009	0.647046115409421
F11	1.300000000000000e+010	1.095447189578642	1.000000000000000e+010	1.167527944425209
F12	1.000000000000000e+010	2. 475735361799080	1.000000000000000e+010	1.257638143994151
F13	1.156261555600451	1.937498420682251	1.208696388185930e+004	0.910749881706651
F14	1.000000000000000e+010	0.671277061080557	1.000000000000000e+010	0.431168318976144
F15	3.229286417411150	0.945858903815962	0.657118204978861e+009	0.199051380299188

Table 4: Comparing CS Algorithm Against CRB in 3D and Engineering Models for Number of Nests=25 (Total Number of Iterations=5000, Time=100)

Function	CS		CRB	
	fmin	Mean Best Nest	fmin	Mean Best Nest
F4	1.200000000000000e+010	2.957017834544342	1.000000000000000e+010	1.674673521443532
F9	1.900000000000000e+010	3.992116826916045	0.602811434499159	3.766666666666667
Spring	1.000000000000000e+010	5.361519719525025	0.393036466676572	0.674020005115085
Speed	3.2000000000000e+010	7.832525695537344	2.362038817325554e+003	6.185714285714285

Beam	1.724852313132572	3.229642967361158s	0.309403299807856	0.100000000000000

5. CONCLUSIONS

The results of nesting and minimum function values in the Tables 1-4 showed that functions (1-18) have a more appropriate reduction than the use of the Cuckoo algorithm alone on these functions. We have reviewed through the tables that when reducing frequency algorithm performance is better than when increasing the number of repetitions and later we proved that the new method has a higher capacity than the algorithm of the cuckoo alone and its strong impact on engineering functions. Calculations were done using the Matlab 2011 program and with a calculator Core i5.

6. APPENDIX

1-min $f(x) = (x_1 - 3)^2 + (x_2 - 2)^2$

s.t $x_1^2 - x_2^2 + 5 = 0$

$x_1 + 2x_2 - 4 \leq 0$

2-min$f(x) = x_1^2 - x_1x_2 + x_2^2$

s.t $x_1^2 + x_2^2 - 4 = 0$

$2x_1 - x_2 + 2 < 0$

3-min$f(x) = (x_1 - 2)^2 + (x_2 - 1)^2$

s.t $x_1 - 2x_2 = -1$

$-\frac{x_1^2}{4} + x_2^2 + 1 \geq 0$

4-min$f(x) = x_1^3 + 2x_2^2 x_3 + 2x_3$

s.t $x_1^2 + x_2 + x_3^2 = 4$

$x_1^2 - x_2 + 2x_3 \leq 2$

5-min$f(x) = e^1 - x_1x_2 + x_2^2$

s.t $x_1^2 + x_2^2 = 4$

$2x_1 + x_2 \leq 2$

6-min$f(x) = x_1^3 - 3x_1x_2 + 4$

s.t $-2x_1 + x_2^2 = 5$

$5x_1 + 2x_2 \geq 18$

7-min$f(x) = -e^{-x_1-x_2}$

s.t $x_1^2 + x_2^2 - 4$

$x_1 - 1 \geq 0$

8-min$f(x) = -x_1^2 + 2x_1x_2 + x_2^2 - e^{-x_1-x_2}$

s.t $x_1^2 + x_2^2 - 4 = 0$

$x_1 + x_2 \leq 1$

9-$\min f(x) = -x_1 x_2 x_3$

s.t $20 - x_1 \geq 0$

$11 - x_2 \geq 0$

$42 - x_3 \geq 0$

$72 - x_1 - 2x_2 - 2x_3 \geq 0$

10-$\min f(x) = (x_1 - 1)^2 + x_2 - 2$

s.t $x_2 - x_1 = 1$

$x_1 + x_2 \geq 2$

11-$\min f(x) = {x_1}^2 + {x_2}^2$

s.t

. $x_1 - 3 = 0$

$x_2 - 2 \leq 0$

12-$\min f(x) = \frac{1}{4000}({x_1}^2 + {x_2}^2) - \cos(x_1)\cos\left(\frac{x_2}{\sqrt{2}}\right) + 1$

s.t $x_1 - 3 = 0$

$x_2 - 2 \leq 0$

13-$\min f(x) = (x_1 - 2)^2 + \frac{1}{4}{x_2}^2$

s.t $2x_1 + 3x_2 = 4$

$x_1 - \frac{7}{2}x_2 \leq 1$

14-$\min f(x) = -x_1 x_2$

s.t $20x_1 + 15x_2 - 30 = 0$

$\frac{{x_1}^2}{4} + {x_2}^2 - 1 \leq 0$

15-$\min f(x) = {x_1}^4 - 2{x_1}^2 x_2 + {x_1}^2 + x_1 {x_2}^2 - 2x_1 + 4$

s.t ${x_1}^2 + {x_2}^2 - 2 = 0$

$0.25\,{x_1}^2 + 0.75{x_2}^2 - 1 \leq 0$

REFERENCES

1. *Abdelaziz, A. Y., Ali, E. S., (2015). Cuckoo Search algorithm based load frequency controller design for nonlinear interconnected power system, International Journal of Electrical Power & Energy Systems, vol. 73, (pp.632-643), DOI:org/10.1016/j.ijepes.2015.05.050.*

2. *Al-Kawaz, R. Z., Ahmed, H. I., Saaed, H. T., (2018). Using Polyak's Log-Sigmoid Method with Firefly Swarm, International Journal of Enhanced Research in Science, Technology & Engineering, vol. 7, no. 5, (pp.24-39).*

3. *Bertsekas, D. P., (1982). Constrained Optimization and Lagrange Multiplier Methods, Academic Press, New York.*

4. Cagnina, L. C., Esquivel, S. C. and Coello, C. A., (2008). Solving engineering optimization problems with the simple constrained particle swarm optimizer, Informatica, vol. 32, (pp.319-326).

5. Charalambous, C., (1977). Nonlinear least pth optimization and nonlinear programming, Mathematical Programming, vol. 12, (pp. 195–225).

6. Chen, H. L., Yu, B., Zhou, H. L. and Meng, Z., (2018). Improved Cuckoo Search Algorithm for Solving Inverse Geometry Heat Conduction Problems, Heat Transfer Engineering, Taylor and Francis, vol. 0, no. 0, (pp.1–13).

7. Dorigo M. and Stutzle T., (2004). Ant Colony Optimization, MIT Press.

8. Fister, I. Jr, Fister, D., and Fister, I., (2013). A comprehensive review of cuckoo search: variants and hybrids, International Journal of Mathematical Modelling and Numerical Optimisation, vol. 4, no. 4, (pp.387–409).

9. Gandomi, A. H., Yang, X. S. and Alavi, A. H., (2013). Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems," Eng. Comput., vol. 29, no. 1, (pp. 17–35). DOI: 10.1007/s00366-011-0241-y.

10. Goldberg D.E., (1989). Genetic Algorithms in Search, Optimization and Machine Learning, Reading, Mass. Addison Wesley.

11. Hestenes, M. R., (1969). Multiplier and gradient method, Journal on Optimization Theory and Applications vol. 4, (pp. 303–320).

12. Holland, J. H., (1975). Adaptation in Natural and Artificial Systems, University of Michigan Press, Ann Arbor.

13. Kaveh, A., Bakhshpoori, T., and Azimi, M., (2015). Seismic optimal design of 3d steel frames using cuckoo search algorithm," Struct. Des. Tall Spec Buildings, vol. 24, no. 3, (pp. 210–227). DOI: 10.1002/tal.1162.

14. Kennedy J. and Eberhart R. C., (1995). Particle swarm optimization, Proc. of IEEE International Conference on Neural Networks, Piscataway, NJ., (pp. 1942-1948).

15. Mangasarian, O.L., (1975). Unconstrained Lagrangians in nonlinear programming, SIAM Journal on Control vol. 13, (pp.772–791).

16. Mareli, M. and Twala, B., (2018). An adaptive Cuckoo search algorithm for optimization, Applied Computing and Informatics, vol. 14, (pp.107-115).

17. Naumann, D. S., Evans, B., Walton, S., and Hassan, O., (2015). A novel implementation of computational aerodynamic shape optimization using modifed cuckoo search," Appl. Math. Model., vol. 40, no. 7, (pp. 4543–4559).

18. Pavlyukevich, I., (2007). Lévy flights, non-local search and simulated annealing," J. Comput. Phys., vol. 226, (pp. 1830–1844). DOI: 10.1016/j.jcp.2007.06.008.

19. Payne, R. B. and Sorensen, M. D., (2005). The Cuckoos. Oxford, UK: Oxford University Press.

20. Polyak, R. A. and Teboulle, M., (1997). Nonlinear rescaling and classical-like methods in convex optimization, Mathematical Programming, vol. 76, (pp.265–284).

21. Polyak, R. A., (1992). Modified barrier function: theory and methods, Mathematical Programming, vol. 54, (pp. 177–222).

22. Polyak, R. A., (2002). Nonlinear rescaling vs. smoothing technique in convex optimization, Mathematical Programming, vol. 92, (pp.197–235).

23. Powell, M. J. D., (1969). A method for nonlinear constraints in minimization problems, in: R. Fletcher (Ed.), Optimization, Academic Press, New York, (pp.283–298).

24. *Rockafellar, R. T., (1973). A dual approach to solving nonlinear programming problems by unconstrained optimization, Mathematical Programming, vol. 5, (pp.354–373).*
25. *Rockafellar, R. T., (1974). Augmented Lagrange multiplier functions and duality in nonconvex programming, SIAM Journal on Control, vol. 12, (pp.268–285).*
26. *Srivatsa, P., Baby, R., and Balaji, C., (2015). Geometric optimization of a PCM-based heat sink–a coupled ANN and GA approach," Heat Transfer Eng., vol. 37, no. 10, (pp.875–888). DOI: 10.1080/01457632.2015.1089749.*
27. *Valian, E., Tavakoli, S., Mohanna, S. and Haghi, A., (2013). Improved cuckoo search for reliability optimization problems," Comput. Indust. Eng., vol. 64, no. 1, (pp. 459–468). DOI: 10.1016/j.cie.2012.07.011.*
28. *Vanderplaats, G. N., (1999). Numerical Optimization Techniques for Engineering Design-With Applications, McGraw-Hill, 3rd edition published by VR&D.*
29. *Vanderplaats, G. N., (2002). Very Large Scale Optimization, Vanderplaats Research and Development, Inc., Colorado Springs, Colorado, NASA/CR-2002-211768 McGraw-Hill.*
30. *Yang, X. S. and Deb, S., (2009). Cuckoo search via lévy flights, presented at the World Congress on Nature and Biologically Inspired Computing, Coimbatore, India, December 9–11, 2009. IEEE Publications, (pp.210–214).*
31. *Yang, X. S. and Deb, S., (2010). Engineering optimization by cuckoo search, Int. J. Math. Model. Numer. Optim., vol. 1, no. 4, (pp.330–343).*
32. *Yang, X. S., (2009). Harmony search as a metaheuristic algorithm. In: Geem Z (ed) Music-inspired Harmony search algorithm. Springer-Verlag, (pp.1–14).*
33. *Agarana, M. C., & Ehigbochie, A. I. (2015). Optimization of Student's Academic Performance in a World-Class University Using Operational Research Technique. International Journal of Mathematics and Computer Applications Research, 5(1), 43-50.*